last project–Practical Machine Learning

Haibin Qian

Background

Based on raw data on accelerometers, my goal is to build models on train data, test performace and predict test data

Tidymodels for this assignment

tidymodels.org

load required libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 0.2.0 --
```

```
## v broom        0.8.0     v rsample      0.1.1
## v dials        0.1.1     v tune         0.2.0
## v infer        1.0.0     v workflows    0.2.6
## v modeldata    0.1.1     v workflowsets 0.2.1
## v parsnip      0.2.1     v yardstick    0.0.9
## v recipes      0.2.0
```

```
## -- Conflicts ---------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Use suppressPackageStartupMessages() to eliminate package startup messages
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following objects are masked from 'package:yardstick':
##
##     precision, recall, sensitivity, specificity
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(skimr)
```

load data sets

```
train_dt <- read.csv(url('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
))

test_dt <- read.csv(url('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'))
```

explory data analysis using skim

```
skim(train_dt) %>%
  tibble::as_tibble() %>%
  dplyr::filter(n_missing != 0) %>%
  .[, 2:4]
```

```
## # A tibble: 67 x 3
##    skim_variable        n_missing complete_rate
##    <chr>                    <int>         <dbl>
##  1 max_roll_belt            19216        0.0207
##  2 max_picth_belt           19216        0.0207
##  3 min_roll_belt            19216        0.0207
##  4 min_pitch_belt           19216        0.0207
##  5 amplitude_roll_belt      19216        0.0207
##  6 amplitude_pitch_belt     19216        0.0207
##  7 var_total_accel_belt     19216        0.0207
##  8 avg_roll_belt            19216        0.0207
##  9 stddev_roll_belt         19216        0.0207
## 10 var_roll_belt            19216        0.0207
## # ... with 57 more rows
```

As can be seen that there are 67 variables possess so many missing data (19216), which is unlikely to be useful for modeling

remove these useless variables

```
uless <- skim(train_dt) %>%
  tibble::as_tibble() %>%
  dplyr::filter(n_missing != 0) %>%
  .[ ,2]

new_train <- train_dt %>%
  select(-as.vector(t(uless))) %>%
  select(-c(1:6))

new_test <- test_dt %>%
  select(-as.vector(t(uless))) %>%
  select(-c(1:6, ncol(.)))
```

remove near zore variance which is also useless for model

```
nzv <- nearZeroVar(new_train)
new_train <- new_train[,-nzv]
new_train$classe <- as.factor(new_train$classe)
new_train <- new_train %>%
  mutate_if(is.integer, as.numeric)
new_test <- new_test[,-nzv]
```

data splitting using rsample from tidymodels

```
set.seed(123)
data_split <- initial_split(new_train, prop = 3/4)
train_train <- training(data_split)
train_test <- testing(data_split)
```

preprocess data and create recipes

```
rec <-
  recipe(classe ~ ., data = train_train)
```

build the models using parsnip in tidymodels – decision_tree first, then random foresst

dt–decision tree

rf–random forest

```
mod_dt <- decision_tree() %>%
  set_engine('rpart') %>%
  set_mode('classification')
```

```
mod_rf <- rand_forest() %>%
  set_engine('ranger') %>%
  set_mode('classification')
```

workflow

```
wfow_dt <-
  workflow() %>%
  add_model(mod_dt) %>%
  add_recipe(rec)

wfow_rf <-
  workflow() %>%
  add_model(mod_rf) %>%
  add_recipe(rec)
```

fit resampled train data set

```
fit_dt <- wfow_dt %>%
  fit(train_train)

fit_rf <- wfow_rf %>%
  fit(train_train)
```

predict

```
aug_dt <- augment(fit_dt, train_test)
aug_rf <- augment(fit_rf, train_test)
```

generate the area under the ROC curve to estimate accuracy

```
aug_dt %>%
  roc_auc(classe, .pred_A:.pred_E)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc hand_till      0.893
```

```
aug_rf %>%
  roc_auc(classe, .pred_A:.pred_E)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc hand_till       1.00
```

As we can see that random forest possess higher accuracy

(However, such high accuracy could attribute to non-resampled train data)

Finally, I decide to use random forest (fit_rf) to predict test data with 20 obs

```
predict(fit_rf, test_dt)
```

```
## # A tibble: 20 x 1
##    .pred_class
##    <fct>
##  1 B
##  2 A
##  3 B
##  4 A
##  5 A
##  6 E
##  7 D
##  8 B
##  9 A
## 10 A
## 11 B
## 12 C
## 13 B
## 14 A
## 15 E
## 16 E
## 17 A
## 18 B
## 19 B
## 20 B
```